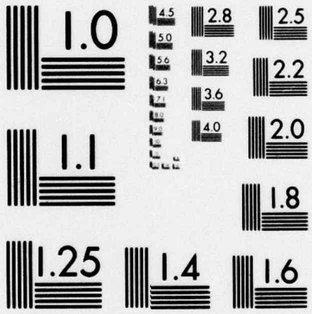
AD-A073 370 YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE F/G 9/2
THE TRANSFER OF INFORMATION AND AUTHORITY IN A PROTECTION SYSTE--ETC(U)
JUL 79 M BISHOP, L SNYDER N00014-75-C-0752
UNCLASSIFIED RR-166 NL

| OF |
AD
A073370

END
DATE
FILMED
10-79
DDC


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

LUX ET VERITAS

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

12

The Transfer of Information and Authority in a Protection System

Matt Bishop

Lawrence Snyder

Research Report #166, July 1979

This research was sponsored in part by the Office of Naval Research Grant N00014-75-C-0752 and by National Science Foundation Grant MCS77-12517.

9 Research rept.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
166

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
6 The Transfer of Information and Authority in a Protection System

5. TYPE OF REPORT & PERIOD COVERED
9 Technical, rept.

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
10 Matt/Bishop ~~and~~ Lawrence/Snyder

8. CONTRACT OR GRANT NUMBER(s)
15 N00014-75-C-0752

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Yale University
Department of Computer Science
10 Hillhouse Ave., New Haven, CT 06520

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14 RR-166

11. CONTROLLING OFFICE NAME AND ADDRESS
Office of Naval Research
Information Systems Program
Arlington, VA 22217

12. REPORT DATE
11 July 1979

13. NUMBER OF PAGES
29
12 34p.

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Distribution of this report is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
information
transfer
Take-Grant Model
de jure
de facto
explicit edge
implicit edge
right
acquisition

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

We distinguish between the transfer of information and the transfer of authority, and extend the Take-Grant Model to describe the former. The conditions under which information (but not authority) might be transferred are given, one characterization using the rules for transferring information alone, and another combining them with the previously-developed rules for transferring authority.

407 051

JCB

1. *Introduction*

Although formal models have contributed to our understanding of capability-based protection systems, they have been properly criticized for concentrating on the movement of "authority" or "access privilege" within the system, rather than on the movement of the information. For example, the Take/Grant Model [1,2] describes the exact conditions under which a particular user can get the authority to access a file. If the conditions are satisfied, then the user can access the information. But if they are not satisfied, it *does not* follow that the user cannot get at the information. There *may be* some way to transfer the information without the user ever getting direct authority to access it. The Take/Grant Model gives no information and other models are similarly mute.

In this paper we take a modest step towards elucidating the problem. Specifically, we distinguish between two types of information acquisition*:

de jure (DJ) acquisition means a user acquires information by invoking direct authority within the capability system;

de facto (DF) acquisition means a user acquires information, usually with the assistance of others, without necessarily acquiring the direct authority to access it within the capability system.

Thus, *de jure* acquisition implies *de facto* acquisition, but not vice versa.

This distinction can be illustrated diagramatically. In Figure 1a, the users have read and write capabilities (r,w) to their personal files. User Abel also has "take" authority over user Baker. This latter authority allows Abel to take the read access authority to File 2 from Baker -- an action that would result in the diagram shown in Figure 1b. Abel can now

*Our use of *de jure*, "rightful, by right" [5] and *de facto* "(existing) in fact, whether by right or not" [5] is intended to avoid the pejorative connotations of the authorized/unauthorized distinction.

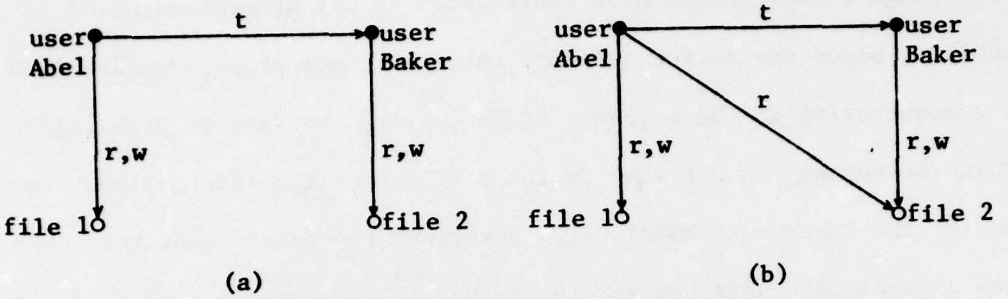


Figure 1: A *de jure* acquisition.

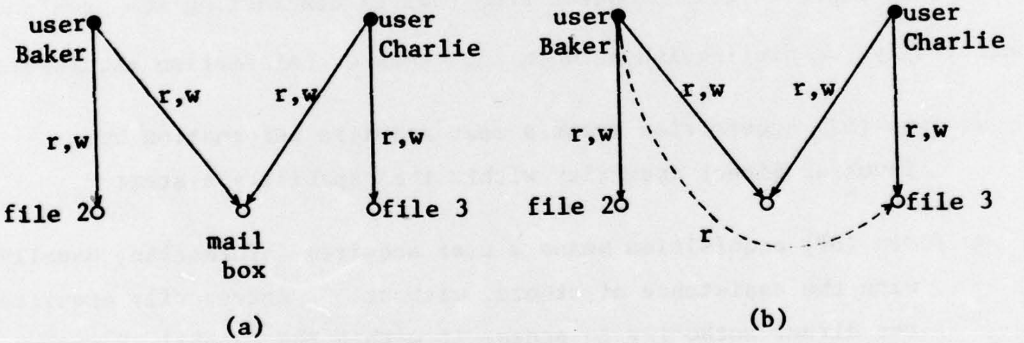


Figure 2: A *de facto* acquisition.

invoke this read authority resulting in a *de jure* acquisition.

Figure 2 illustrates a situation when two users have "read" and "write" capabilities to their personal files as well as a common mail box file. Baker can request that user Charlie write the information from File 3 into the mail box. Assuming Charlie complies fully, Baker can then read the information from the mail box resulting in *de facto* acquisition. Baker never has the "read" capability to File 3 although he can read a copy of it. Having the capability to read a file and being capable of reading a copy of it are not the same thing because (a) the latter relies on the transmission of a complete and accurate copy and (b) any updates to File 3 are not automatically reflected in the copy. We use a dashed line to denote the *de facto* transmission.

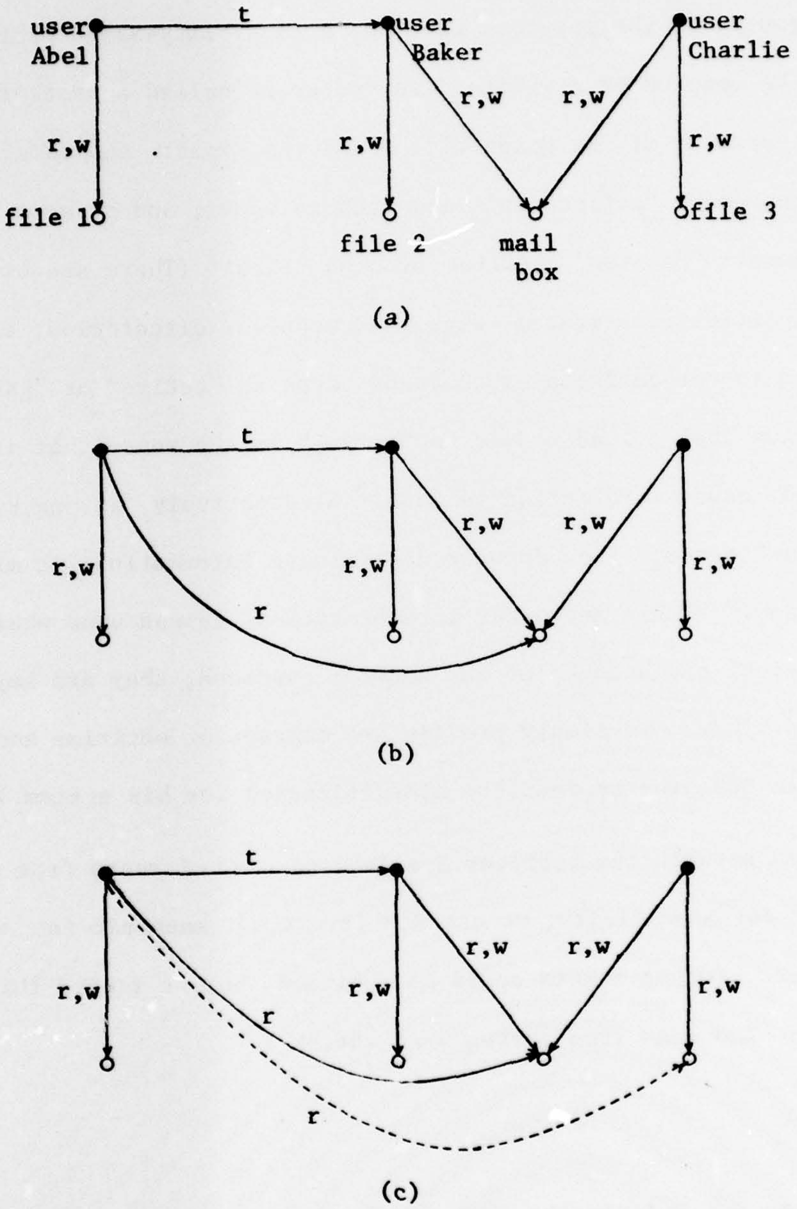
Obviously, more complex situations can arise. In the graph formed by combining Figures 1a and 2a, we can illustrate both types of transfer. In Figure 3, Abel takes the "read" capability to the mail box. Then, after Charlie writes File 3 into the mail box, perhaps in the belief that he is making it available only to Baker, Abel can make a *de facto* acquisition.

Our objective in this paper is to characterize the use of *de facto* and *de jure* acquisition in a protection system. Since *de jure* acquisition is already well understood in the Take/Grant Model, we build on that understanding to develop conditions under which information can be transferred by *de facto* transfers only or by a combination of *de facto* and *de jure* transfers.

We shall organize our presentation as follows. (Note that no previous knowledge of the Take/Grant System is presumed.)

Section 2: Definition of the model and the class of *de facto* rules,
Section 3: Characterization of *de facto* acquisition,
Section 4: *De jure* acquisition and previous results,
Section 5: Characterization of combined *de facto* and *de jure* acquisition.

The final section is devoted to a summary and discussion.



(a)

(b)

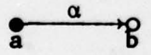
(c)

Figure 3: Combination of *de jure* and *de facto* transfers.

2. De Facto *Information Transfers*

As suggested in the previous section, a capability-based protection system will be modeled by a finite directed graph called a *protection graph*. The vertices of the graph will be of two types: *subjects* (denoted by ●) will represent "active" entities such as users, and *objects* (denoted by ○) will denote "passive" entities such as files. (There are ususally many other entities in a system, e.g. load modules, directories, etc., that are hard to categorize by such vague terms as "active" or "passive." One might argue that a load module is "active" in the sense that it could, when executed, cause information to move. Alternatively, if one knows that the module is "secure," i.e. doesn't disseminate information, it might be called "passive." These and other interpretations depend upon what system is being modeled, and because of our general approach, they are beyond the scope of this study. We simply provide two classes of entities and depend on the user to make the appropriate classification for his system.)

The edges between the vertices are labeled with elements from a finite set R of *rights*. For specificity, we use $R = \{r,w,t,g\}$, mnemonic for *r*ead, *w*rite, *t*ake and *g*rant. (Other rights could be included, but we regard this as a minimal set.) The edge from vertex a to vertex b

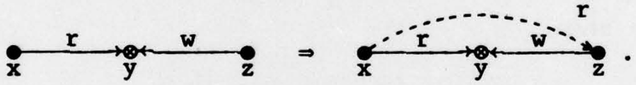


labeled by some $\alpha \subseteq R$ indicates that within the protection system, a has the α rights to b. This edge is called an *explicit* edge.

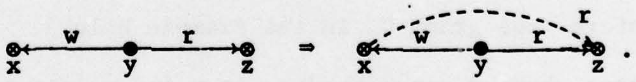
In addition to these solid edges, we will use dashed edges (labeled by an r) to represent *de facto* acquisitions. These edges are called *implicit* edges. They are not actually part of the protection graph since they represent information that is not part of the protection system. But we add them to the protection graph for pedogogical reasons to specify the existence of a potential *de facto* transfer.

In the Introduction we illustrated how information might be transferred in a system by means of a mail box construction, but may be other means as well. We identify four distinct methods of *de facto* acquisition and formulate them as rewriting rules on protection graphs. (Note, in the following definitions "edge" refers to either an explicit or implicit edge. In the diagrams, ⊗ denotes a vertex that can be either a subject or object, edge labels may contain additional rights, set braces are elided.)

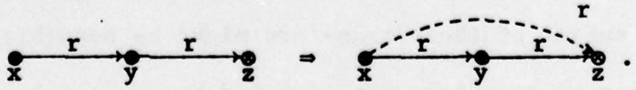
Post: Let x, y and z be distinct vertices of a protection graph G such that x and z are subjects. Let there be an edge from x to y labeled α, $r \in \alpha$, and an edge from z to y labeled β, $w \in \beta$. Then *post* defines a new graph G' with an implicit edge from x to z labeled r. Graphically,



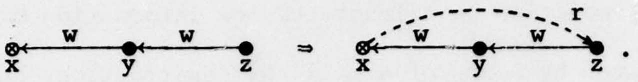
Pass: Let x, y and z be distinct vertices in a protection graph G such that y is a subject. Let there be an edge from y to x labeled by α, $w \in \alpha$, and an edge from y to z labeled by β, $r \in \beta$. Then *pass* defines a new graph G' with an implicit edge from x to z labeled r. Graphically,



Spy: Let x, y and z be distinct vertices in a protection graph G such that x and y are subjects. Let there be an edge from x to y labeled α, $r \in \alpha$, and an edge from y to z labeled β, $r \in \beta$. Then the *spy* rule defines a new graph G' with an implicit edge from x to z labeled r. Graphically, we write



Find: Let x, y and z be distinct vertices in a protection graph G such that y and z are subjects. Let there be an edge from y to x labeled α, $w \in \alpha$, and an edge from z to y labeled β, $w \in \beta$. Then *find* defines a new graph G' with an implicit edge from x to z labeled r. Graphically,



We will refer to these rules, collectively, as the DF rules.

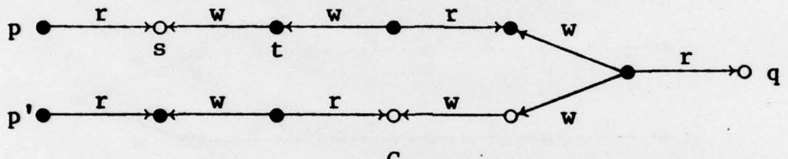
The rules are intended to abstract possible ways in which information can be read in a system by the cooperative effort of one or more subjects. The subjects invoke authority that they own within the system (*de jure* acquisition) in order to effect *de facto* transfer. This transfer, or more accurately, the potential for this transfer, is summarized by the implicit edge from x to z, labeled r. We can then apply these rules to a protection graph (see example below) to summarize the *de facto* transfer in the entire system.

Clearly, the Post rule abstracts the operation described in the Introduction. In the Pass rule y acts as a conduit through which data travels from z to x. The Spy rule abstracts the case where y reads data from z and x "watches" y read the data. More often, however, it is used to "compose" transfers (see graph G_5 in the example below). The Find rule abstracts the case where z deposits data in y and y in turn passes it along to x.

We regard these four rules as a representative sample of the potential *de facto* transfers that might arise in a protection system. In some actual systems only a subset of these transfers might be possible while in other systems there may be transfers not captured by these rules. In either case the development that follows may have to be modified. Our purpose is to illustrate how the Take/Grant Model can be used to assess the potential *de facto* transfers of a protection system.

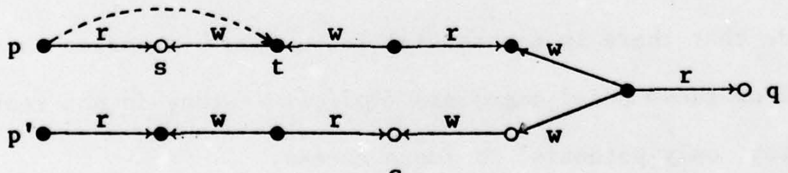
Finally, note that we have concerned ourselves only with the transfer of information to x via read. We might also have considered transmission of information *from* x by the addition of rules that add edges labeled with a "w." We shall discuss this apparent limitation in Section 6.

The rewriting rules enable us to illustrate the potential *de facto* transfers by augmenting a given protection graph G with new implicit edges. Let G_0 be the protection graph



G_0

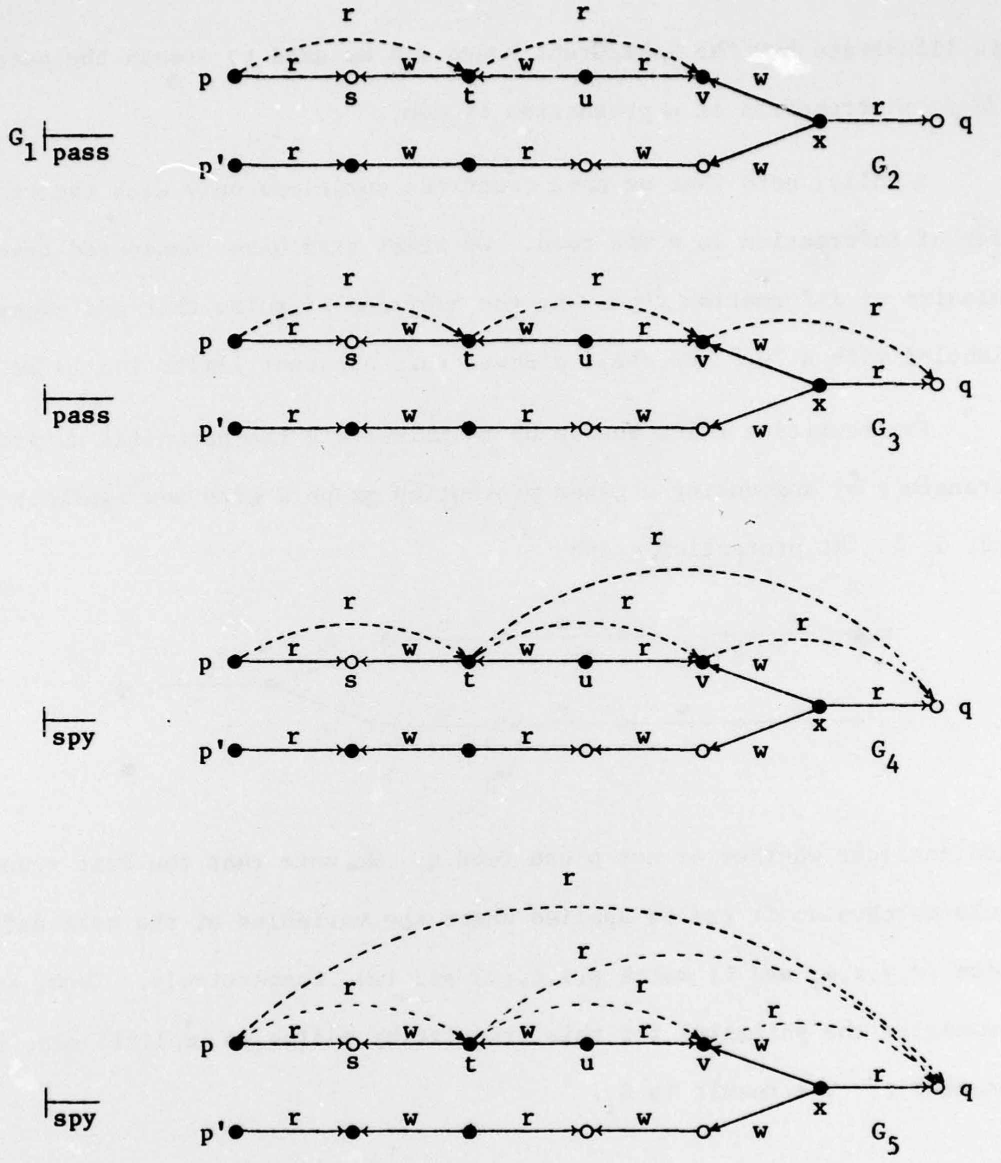
and consider whether or not p can read q. We note that the Post transfer rule matches so it can be applied where the variables of the rule definition (x,y,z,α and β) match p,s,t,{r} and {w}, respectively. Thus, we summarize the potential for this transfer by adding an implicit edge from p to t labeled r. The result is G_1.



G_1

Usually, we denote such a rule application by $G_0 \vdash_{\text{post}} G_1$.

The sequence of rule applications that illustrate that p could acquire the contents of q are illustrated below.

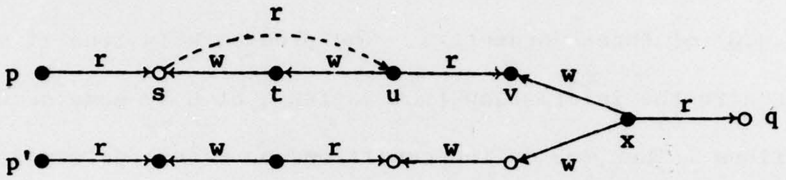


So we conclude that there is a potential for *de facto* transfer to p. Note that all of these added edges are *implicit* -- they do not represent added authority, only potential *de facto* access.

Tortuous though the example may be, it illustrates that rather complex transfers can be realized. It is just as important (perhaps more important) to know what *de facto* transfers cannot be realized. For example, it is *not* possible for p' to read q by a transfer along the "lower" path in G_0.

This is because of the two consecutive objects which form a "barrier" to indirect transfer. (See Theorem 3.1.)

To illustrate another subtlety, note that t plays a pivotal role in the transfer. We might have tried to skip past t by applying the Find rule to G_0.



But s is an object, and our rule definitions do not permit the application of a Spy to define a read edge from p to u. One might argue that a Spy should be allowed here because the s-to-u read edge is implicit and thus s receives the information passively. Subjecthood appears restrictive. Our decision to force the second vertex in a Spy rule to be a subject guarantees the existence of an agent when needed. It will be clear from our results that this limitation is not serious.

Finally, we must make one cautionary remark concerning the interpretation of protection graphs. This is a general study that will be applicable (we hope) to a wide class of protection systems. As such we must consider all protection graphs even if they do not have a sensible interpretation in the context of a *particular* protection system. For example, we allow constructs such as $\underset{x}{\circ}\xrightarrow{r}\underset{y}{\circ}$ in our protection graphs. If one thinks of objects as files, this may be meaningless. But if objects include "secure" processes, then this is more reasonable. We cannot limit *a priori* the class of interpretations, so we allow for any protection graph consistent with our original definitions.

3. *The Conditions of* De Facto *Transfer*

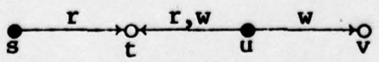
Having abstracted potential *de facto* transfers as a set of four rewriting rules and having illustrated that these rules compose in complex ways, we now formulate an exact statement of what it means for a potential *de facto* transfer to exist within the model. This will be done by defining a predicate *can•know•f*(p,q,G) of three parameters. The predicate is true if vertex p of G can acquire the information from vertex q of G by some sequence of rule applications. Then, we define conditions on G that determine when the predicate is true.

Define for a protection graph G_0 and arbitrary distinct vertices p and q of G_0

> *can•know•f*(p,q,G) to be true if and only if there exists a sequence of graphs $G_1,\ldots,G_n$ $(n \geq 0)$ such that G_{i+1} follows from G_i by one of the DF rules $0 \leq i < n$ and in G_n there is a p-to-q edge labeled r.

Thus, the predicate *can•know•f*(p,q,G_0) is true if and only if *de jure* authority exists or an implicit edge from p-to-q can be added by means of the four DF rules.

Now, we formulate conditions under which *can•know•f* holds. To aid in this endeavor, define an *rw-path* in a protection graph G as a sequence of distinct vertices $v_0,v_1,\ldots,v_k$ $(k > 1)$ such that v_i is connected to v_{i+1} by an edge (in either direction) labeled with r or w (or both) for all i, $0 \leq i < k$. We say that the rw-path is *between* v_0 and v_k. For example, in the graph



the sequence s,t,u,v is an rw-path.

Not all rw-paths will permit *de facto* transfer of information. (For example, s,t,u,v above does not!) So we limit our attention to a certain subset of them. To do this, we associate with each rw-path one or more words over the alphabet $\{\vec{r},\overleftarrow{r},\vec{w},\overleftarrow{w}\}$ in the obvious way; for example, the sequence s,t,u,v given above has associated words, namely $\vec{r}\overleftarrow{r}\vec{w}$ and $\vec{r}\overleftarrow{w}\vec{w}$.

Define an rw-path $v_0,v_1,\ldots,v_k$ $(k > 1)$ to be an *admissible rw-path* if and only if

(i) it has an associated word $a_1a_2 \ldots a_k$ in the regular language $(\vec{r} \cup \overleftarrow{w})^*$ and

(ii) if $a_i = \vec{r}$ then v_{i-1} is a subject and if $a_i = \overleftarrow{w}$ then v_i is a subject.

There are two immediate consequences of this definition. First, since $k > 1$, there are always at least two letters in the word associated with any admissible path. Second, there cannot be two consecutive objects on any admissible path.

The first result concerning *de facto* transfers can now be stated.

Theorem 3.1: Let p and q be vertices in a protection graph G. Then *can•know•f*(p,q,G) is true if and only if there is a p-to-q edge labeled r or there is an admissible rw-path between p and q.

Proof: ($\Leftarrow$) By induction on the length ℓ (i.e. number of edges) of the admissible rw-path.

(Basis): Clearly, when $\ell = 2$ (the shortest non-trivial length) there are four distinct rw-paths and each of these is handled by a separate rule.

(Induction): Let the hypothesis be that for each ℓ, $2 \leq \ell \leq k$, if $p = v_0,v_1,\ldots,v_\ell = q$ is an admissible rw-path then *can•know•f*(p,q,G) is true. Observe that for every admissible rw-path of length $\ell > 2$ either it is an extension by $\vec{r}$ of an admissible rw-path terminating in a subject or it terminates in a subject and is the extension by $\overleftarrow{w}$ of an admissible rw-path. Let $p = v_0,\ldots,v_k,v_{k+1} = q$ be an admissible rw-path. By hypothesis *can•know•f*(p,v_k,G) is true and hence a p-to-v_k edge labeled $\vec{r}$ can be constructed. By the observation either a Spy or Post rule can be applied to give a p-to-q edge labeled $\vec{r}$ and *can•know•f*(p,q,G) is true, extending the induction.

($\Rightarrow$) By induction on the number ℓ of times any of the four rules are applied to produce a witness to *can•know•f*.

(Basis): By inspection of the rule schemata, if only one rule is applied then the path between the vertices is an admissible rw-path.

(Induction): Suppose that all witnesses to *can•know•f* requiring $\ell > 1$ or fewer rule applications have admissible rw-paths, and let a witness to *can•know•f*(p,q,G) require ℓ+1 rule applications. Since edges labeled with $\overleftarrow{w}$ cannot be introduced, the $\ell+1^{st}$ rule could not have been the Find rule. If the $\ell+1^{st}$ rule was a Pass or Post rule then the edge of the rule schema labeled $\overleftarrow{w}$ is explicit and the edge labeled $\vec{r}$ was constructed between, say, x and y with ℓ rule applications. Then *can•know•f*(x,y,G) is true and by hypothesis there is an admissible rw-path between x and y. The extension of this path by w leads, by inspection, to an admissible path. Finally, if the $\ell+1^{st}$ rule was a Spy then there are edges labeled $\vec{r}$ between some x and y, and y and z. If one of these is explicit, say the x-to-y edge, then *can•know•f*(y,z,G) is true and the edge found in ℓ rule applications. By hypothesis there is an admissible rw-path between y and z and by inspection the extension is an admissible rw-path between x and z. If both the x-to-y and y-to-z edges are implicit then by analogous reasoning they are admissible. Since the concatenation of admissible paths is admissible, the induction is extended. □

We emphasize that this condition is necessary and sufficient (i.e. if and only if); it exactly characterizes the way DF rules can cause *de facto* transfers. It is also clear that using standard breadth-first graph traversal techniques, this condition is easy to test, for any given pair of vertices.

> *Corollary 3.2:* For vertices p and q of a protection graph G, there is a linear-time (in the size of the graph), algorithm for testing *can·know·f*(p,q,G).

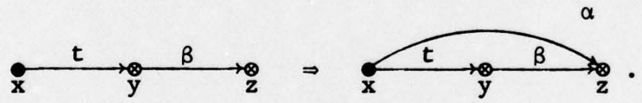
The reader is encouraged to return to the graph G_0 in Section 2 to verify our claim that there can be no transfer along the "lower" path; that is, *can·know·f*(p',q,G_0) is false.

4. *Review of* De Jure

Up to this point we have concentrated on the four rules that implement *de facto* transfers. Although these rules specify the addition of an edge in the graph, we have agreed that these are only *implied* edges -- no new access authority has been created. Now, we review the way in which *de jure* acquisition takes place in the Take/Grant Model.

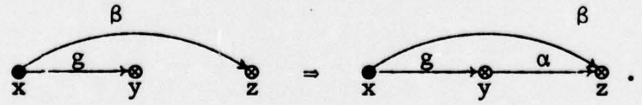
Recall that in addition to r and w, there are two other rights: t and g. In [1,2] the following rules were introduced for changing access authority. All edges referred to in these rules are explicit.

Take: Let x, y and z be three distinct vertices in a protection graph G such that x is a subject. Let there be an edge from x to y labeled γ such that $t \in \gamma$, an edge from y to z labeled β and $\alpha \subseteq \beta$. Then the *take* rule defines a new graph G' by adding an edge to the protection graph from x to z labeled α. Graphically,



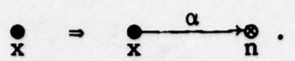
The rule can be read: "x takes (α to z) from y."

Grant: Let x, y and z be three distinct vertices in a protection graph G such that x is a subject. Let there be an edge from x to y labeled γ such that $g \in \gamma$, an edge from x to z labeled β, and $\alpha \subseteq \beta$. The *grant* rule defines a new graph G' by adding an edge from y to z labeled α. Graphically,



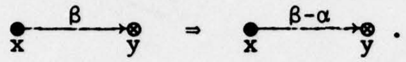
The rule can be read: "x grants (α to z) to y."

Create: Let x be any subject vertex in a protection graph G and let α be a subset of R. *Create* defines a new graph G' by adding a new vertex n to the graph and an edge from x to n labeled α. Graphically,



The rule can be read: "x creates (α to) new $\left\{\begin{matrix}\text{subject}\\ \text{object}\end{matrix}\right\}$ n."

Remove: Let x and y be any distinct vertices in a protection graph G such that x is a subject. Let there be an edge from x to y labeled β, and let α be any subset of rights. Then *remove* defines a new graph G' by deleting the α labels from β. If β becomes empty as a result, the edge itself is deleted. Graphically,



The rule can be read: "x removes (α to) y."

We refer to these four rules collectively as the *DJ rules.*

The edges added by these rules represent explicit changes in the access authority. Thus, when "x takes (r to z) from y," x only acquires the read rights to the information. It must invoke the right to read the information. In addition to adding edges, Create allows the addition of new vertices. As Figure 4 illustrates,* Create adds an important dimension to the model since without Create one cannot add g to the *a-to-b* edge in this example.

In order to report on previous results [1,2] we define *tg-path* (analogous to an rw-path) as a nonempty sequence $v_0,\ldots,v_k$ of vertices such that for all i, $0\leq i<k$, v_i is connected to v_{i+1} by an edge (in either direction) with a label containing a t or g (or both). Vertices are *tg-connected* if there is a tg-path between them and we call any maximal, tg-connected subject-only subgraph an *island.*

Associate with tg-paths words over the alphabet $\{\vec{t},\overleftarrow{t},\vec{g},\overleftarrow{g}\}$ analogous to the words associated with rw-paths. (If k=1 in the tg-path, then the

*Note, even though there is only one directed edge from any vertex a to any vertex b, we occasionally draw two to emphasize changes in labelling.

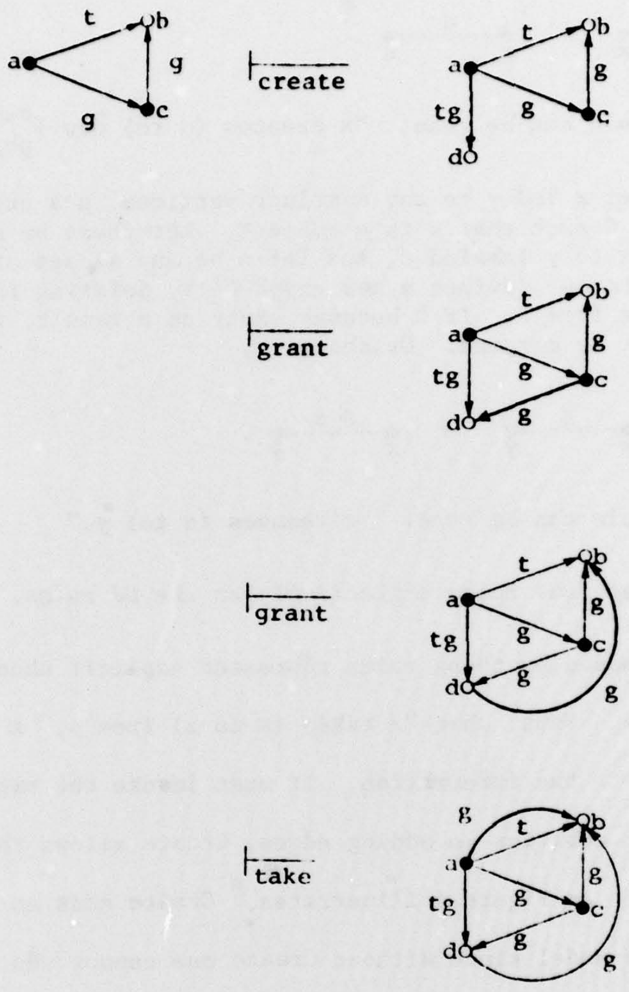


Figure 4: Vertex a acquires g rights to b, i.e., g is added to the label on the a-to-b edge. The rule applications may be read:

a creates (tg to) new object d,

a grants (g to d) to c,

c grants (g to b) to d,

a takes (g to b) from d.

associated word in ϵ.) A tg-path $v_0,\ldots,v_k$ with v_0 being a subject is an *initial span* if it has an associated word in the language $\{\vec{t}^*\vec{g}\} \cup \{\epsilon\}$; it is a *terminal span* if it has an associated word in $\{\vec{t}^*\}$; and it is a *bridge* if v_k is a subject and it has an associated word in $\{\vec{t}^*, \overleftarrow{t}^*, \vec{t}^*\vec{g}\overleftarrow{t}^*, \vec{t}^*\overleftarrow{g}\overleftarrow{t}^*\}$. Note that initial and terminal spans have orientation, i.e. v_0 is the *source* of the spans. We say v_0 initially or terminally spans to v_k.

Restricting our attention only to Take, Grant, Create and Remove, we define for a right α and distinct vertices p and q of a protection graph G_0, the predicate

> *can·share*$(\alpha,p,q,G) \Leftrightarrow$ there are protection graphs $G_1,\ldots,G_n$ such that $G_0 \overset{*}{\vdash} G_n$ using only DJ rules and in G_n there is p-to-q edge labeled α.

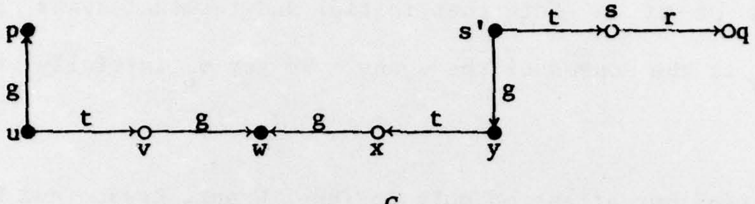
Note that α can be any right in $R = \{r,w,t,g\}$.

We may now state when the *can·share* predicate is true. Let p and q be arbitrary, distinct vertices in protection graph G_0 and let $\alpha \in R$.

> *Theorem 4.1* [2]: The predicate *can·share*(α,p,q,G_0) is true if and only if the following hold simultaneously:
>
> (i) there is a vertex $s \in G_0$ with an s-to-q edge labeled α,
>
> (ii) there exist subject vertices p' and s' such that
>
> (a) p' initially spans to p,
>
> (b) s' terminally spans to s,
>
> (iii) there exist islands $I_1,\ldots,I_v$ and there is a bridge from I_j to I_{j+1} $(1 \le j < v)$.

Figure 5 illustrates the conditions of the theorem. Although these conditions appear to be complicated, we can test a protection graph in linear time to see if it satisfies the conditions.

Clearly, if one is restricted to the DJ rules, then p can get *de jure* access to q in G_0 if and only if *can•share*(r,p,q,G_0) is true. The crucial question is: how do the DJ and DF rules interact? We describe that in the next section.



G_0

Islands: $I_1 = \{p,u\}$, $I_2 = \{w\}$, $I_3 = \{y,s'\}$.

Bridges: u,v,w and w,x,y.

Initial span: p; associate word: ϵ.

Terminal span: s',s; associated word: $\vec{t}$.

Can•share(r,p,q,G_0) is true as the following rules attest.

1. s' takes (r to q) from s.
2. s' grants (r to q) to y.
3. y takes (g to w) from x.
4. u takes (g to w) from v.
5. u grants (g to p) to w.
6. y grants (r to q) to w.
7. w grants (r to q) to p.

The resulting graph appears as follows:

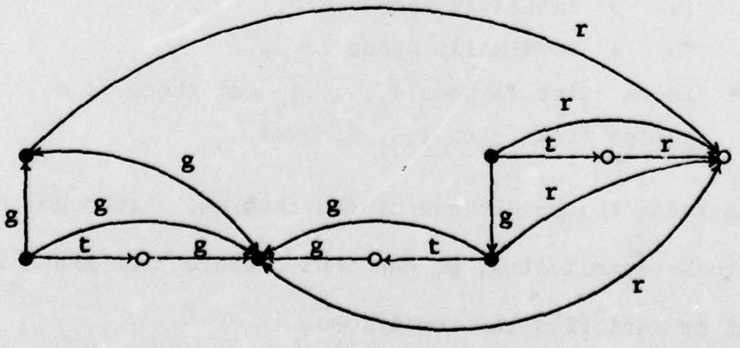
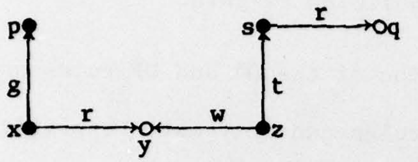


Figure 5: Illustration of the conditions of *can•share*.

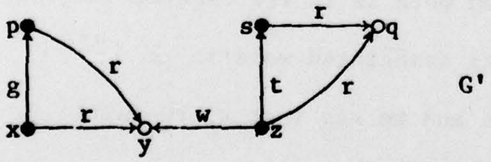
5. *Combined transfers*

We begin by illustrating a simple case where both *de jure* and *de facto* transfers are needed to share information. Consider the protection graph G

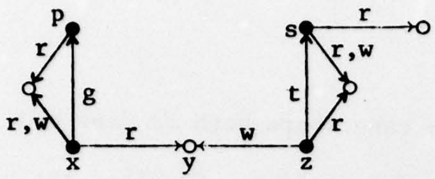


and notice that *can•share*(r,p,q,G) is false since s (the only owner of the read right to q) is not tg-connected to p. Also, *can•know•f*(p,g,G) is false since there is no admissible rw-path between p and q. Furthermore, by our Theorem 4.1, no matter what changes we make to G using Take, Grant, Create and Remove, *can•share*(r,p,q,G) remains false, and by our Theorem 3.1 no matter what changes we make to G using Spy, Post, Pass and Find, *can•know•f*(p,q,G) remains false. But, it is possible using DJ and DF rules to construct a graph G' in which *can•know•f*(p,q,G) is true.

In fact, there are two ways to change the graph that are conceptually different. First, x can grant (r to y) to p and z can take (r to q) from s. This results in the graph G'



which now contains an admissible rw-path. Alternatively, in G x and s can create r,w rights to new objects and "read" rights to these objects can be acquired by p and z to "straddle" the t and g edges. The result is G''



G''

which contains an admissible rw-path. Thus, we can either transmit existing rights or create new rights to build an rw-path.

We refer to the use of any combination of the DJ and DF rules as *combined transfer*. (Recall that the DJ rules can only match explicit edges while the DF rules can match explicit or implicit edges.)

Following our paradigm, we define a predicate that introduces a read edge by any of the combined transfers. Let p and q be arbitrary, distinct vertices in a protection graph G_0, then

> *can•know*(p,q,G_0) is true if and only if there is a sequence of protection graphs $G_1,\ldots,G_n$ such that $G_0 \vdash^{*} G_n$ and in G_n there is a p-to-q edge labeled r.

Note that the p-to-q edge can be either implicit or explicit.

Define *rwtg-path* in the obvious way and associate words over the alphabet $\{\overrightarrow{t},\overleftarrow{t},\overrightarrow{g},\overleftarrow{g},\overrightarrow{r},\overleftarrow{r},\overrightarrow{w},\overleftarrow{w}\}$ as usual. We define a second class of spans. Let $v_0,\ldots,v_k$ $(k > 0)$ be an rwtg-path where v_0 is a subject. This path is an *rw-initial span* if its associated word is in the regular language $\{\overrightarrow{t}^{*}\overrightarrow{w}\}$ and it is an *rw-terminal span* if its associated word is in $\{\overrightarrow{t}^{*}\overrightarrow{r}\}$. Again we observe that spans have orientation and we say that v_0 rw-initially (or rw-terminally) spans to v_k.

Define the regular languages:

Bridges: $B = \{\overrightarrow{t}^{*} \cup \overleftarrow{t}^{*} \cup \overrightarrow{t}^{*}\overrightarrow{g}\overleftarrow{t}^{*} \cup \overrightarrow{t}^{*}\overleftarrow{g}\overleftarrow{t}^{*}\}$,

Connections: $C = \{\overrightarrow{t}^{*}\overrightarrow{r} \cup \overleftarrow{w}\overleftarrow{t}^{*} \cup \overrightarrow{t}^{*}\overrightarrow{r}\overleftarrow{w}\overleftarrow{t}^{*}\}$.

Note that the bridges language is the same set defined in Section 4.

We can now characterize the *can•know* predicate. Let p and q be arbitrary, distinct vertices in a protection graph G.

Theorem 5.1: *can•know*(p,q,G) is true if and only if

(i) *can•share*(r,p,q,G) is true or,

(ii) there exists a sequence of subjects $u_1,\ldots,u_n$ such that the following conditions hold:

(a) $p = u_1$ or u_1 rw-initially spans to p,

(b) $q = u_n$ or u_n rw- terminally spans to q, and

(c) for all i, $1 \le i < n$ there is an rwtg-path between u_i and u_{i+1} with associated word in $B \cup C$.

Proof: (⇒) If *can•know*(p,q,G) is true and a witness can be found by application of DJ rules only then obviously *can•share*(r,p,q,G) is true. So suppose that at least one application of a DF rule is required to construct a witness G_w for *can•know*(p,q,G). Because DJ rules do not manipulate implicit edges, we can without loss of generality, arrange the rule applications so that all DJ rules are performed before any DF rules are applied. Let G_j denote the protection graph resulting from the application of only DJ rules. Further, note that among the DJ rules, all Creates can be performed before any of the Take, Grant or Remove* rules. Let G_c denote the result of applying all Creates to G. Clearly, the following relations hold among the graphs.

$$G \vdash^{*}_{\substack{\text{Create}\\ \text{only}}} G_c \vdash^{*}_{\substack{\text{other}\\ \text{DJ}\\ \text{rules}}} G_j \vdash^{*}_{\substack{\text{only}\\ \text{DF}\\ \text{rules}}} G_w.$$

Next, notice that each of the newly created vertices in G_c is in a created subgraph that is connected to the G subgraph of G_c via exactly one original subject vertex. If v is a created vertex, call this subject

*Clearly, Remove rule applications are never useful in this context since additional edges are not harmful.

vertex the *father* of v. (Of course, the father need not have actually created v, but it must have created one of the vertices in the created subgraph in which v resides.)

Since only DF rules are applied after the creation of G_j, it follows by Theorem 3.1 that there exists in G_j an admissible rw-path $p = v_0, v_1, \ldots, v_k = q$ between p and q. We shall reason about how this path was constructed by means of the DJ rules.

The following three facts, derived from Theorem 4.1, will be helpful in the argument. Suppose for arbitrary, distinct vertices x and y in a protection graph G' *can•share*(r,x,y,G') (resp. *can•share*(w,x,y,G') is true.

Fact 1: Either there is an x-to-y edge in G' labeled r (resp. w) or there is a subject s in G' and an rwtg-path in G' from s to y with associated word in $\{\overrightarrow{t}^{*}\overrightarrow{r}\}$ (resp. $\{\overrightarrow{t}^{*}\overrightarrow{w}\}$).

Fact 2: If a witness can be found using islands $I_1, \ldots, I_t$ then *can•share*(r,z,y,G') (resp. *can•share*(w,z,y,G')) is true for any subject $z \in I_j$, $1 \le j \le t$.

Fact 3: If there is no x-to-y edge labeled r (resp. w) then there is a sequence of subjects $x = w_0, w_1, \ldots, w_m = s$ such that w_j is connected to w_{j+1} by a bridge.

Proceeding with the analysis of the admissible rw-path, let v_i and v_{i+1} be consecutive vertices along the path. Suppose v_i and v_{i+1} are both in G then *can•share*(r,v_i,v_{i+1},G) (resp. *can•share*(w,v_{i+1},v_i,G)) is true. Then Fact 1 assures that they are connected by an edge in C or there is an s connecting to v_{i+1} (resp. v_i) by an rwtg-path in C. If v_i and v_{i+1} are both subjects, and $s = v_i$ (resp. $s = v_{i+1}$) then v_i and v_{i+1} qualify as subjects u_j and u_{j+1} for some j. If v_i and v_{i+1} are subjects but $s \neq v_i$

(resp. v_{i+1}) then Fact 3 guarantees the existence of bridge connected subjects $v_i = w_0, \dots, w_m = s$ which qualify as $u_j, \dots, u_{j+m}$ for some j and m.

By admissibility, only v_{i+1} (resp. v_i) can be an object. If $v_{i+1} = q$ (resp. $v_i = p$) then Fact 1 guarantees an rw-terminal span (resp. rw-initial span) from s to q (resp. p). Then s qualifies as subject u_n (resp. u_1).

Assume $v_{i+1} \neq q$ (resp. $v_i \neq p$) is an object and let s be defined by Fact 1. By admissibility, the next vertex v_{i+2} (resp. v_{i-1}) must be a subject. Suppose this next vertex is in G. Then *can·share*(w, v_{i+2}, v_{i+1}, G) (resp. *can·share*(r, v_{i-1}, v_i, G) is true and by Fact 1, s' exists connecting to v_{i+1} (resp. v_i) by a word in C. Now s and s' qualify as u_j and u_{j+1} (resp. u_j and u_{j-1}) for some j since they are connected by a word in $\{\vec{t}^{*}\vec{r}\overleftarrow{w}\overleftarrow{t}^{*}\}$. Moreover, by Fact 3 if $v_{i+2} \neq s'$ (resp. $v_{i-1} \neq s'$) there are subjects $s' = w_0, \dots, w_m = v_{i+2}$ (resp. $v_{i-1} = w_0, \dots, w_m = s'$) which are bridge connected and thus qualify as $u_{j+1}, \dots, u_{j+m+1}$ (resp. $u_{j-m-1}, \dots, u_{j-1}$).

Now suppose that one or more vertices v_{i-1}, v_i, v_{i+1}, or v_{i+2} mentioned in the preceding paragraphs are not in G. Then the preceding argument applies without modifications in G_c. In the application of the *can·share* predicate in that argument, the fathers of the new vertices must be in islands witnessing the sharing since these new vertices are connected to the G subgraph via the father. Thus, for example, if v_i is a new vertex and v_{i+1} is an existing vertex and *can·share*(r, v_i, v_{i+1}, G_c) is true, then by Fact 2, *can·share*$(r, \text{father}(v_i), v_{i+1}, G)$ is true. Thus the father(v_i) *acts as a surrogate* for v_i. In particular, the bridges that were shown to exist for v_i in the original argument, must exist for the father of v_i. If v_{i+1} is also a new vertex, both fathers are surrogates and they are connected by bridges over 0 or more islands. The details are left to the reader.

Finally, we observe that for each pair of consecutive vertices, we established the existence of subjects $u_j,\ldots,u_{j+m}$ for some j and m. Since adjacent pairs will have subject sequences with a common element, the existence of the entire sequence has been established.

($\Leftarrow$) If *can•share*(r,p,q,G) is true, *can•know*(p,q,G) is trivially true. So suppose it is false and let $u_1,\ldots,u_n$ be the subjects required in condition (ii) of the Theorem. It is sufficient to convert this to an admissible rw-path and then invoke Theorem 3.1. If u_i and u_{i+1} are connected by a word in C or conditions (a) or (b) apply, then use the Take rule in the obvious way until no further applications are possible. An r or w connecting edge results. Otherwise u_i and u_{i+1} are connected by a bridge. Apply Take in the obvious way until no further applications are possible. Then u_i and u_{i+1} are connected by an edge with word in $\{\vec{t},\overleftarrow{t},\vec{g},\overleftarrow{g}\}$. Now one of the vertices can Create (rw to) a new object and the other can acquire the appropriate right so that u_i and u_{i+1} are connected by a path with a word in $\{\vec{r}\overleftarrow{w}\}$. The result is an admissible rw-path, and Theorem 3.1 can be applied. □

Corollary: For arbitrary, distinct vertices p and q in a protection graph G, the predicate *can•know*(p,q,G) can be tested in linear time in the size of the graph.

Although the proof is quite involved, the conditions are quite straight-forward. The reader is encouraged to return to the graph presented at the beginning of the section to verify that they do apply.

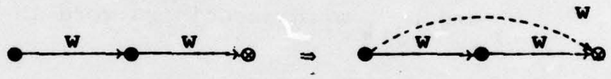
6. *Concluding Remarks*

Two issues remain to be discussed: "two-way" *de facto* transfers and the "worst-case" assumption.

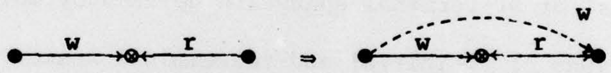
In the foregoing sections we have concerned ourselves with *de facto* transfers in which p can read the contents of q -- a one-way transfer of information. Suppose p would like to communicate back to q, i.e. establish two-way communication. Must we repeat this entire development for the write right? Not at all!

Observe that by interchanging the r and w labels on our DF rule schemata we obtain the following:

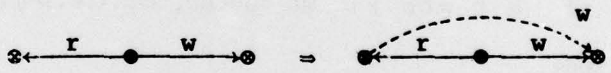
spy-w



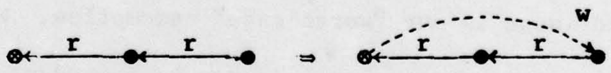
post-w



pass-w



find-w



.

These new DF-w rules reflect the symmetry of read and write and are intuitively consistent.* Moreover, the directionality of the edges and the subject/object distinctions are all preserved. Thus, by interchanging r and w in the foregoing section, all substantive aspects of the arguments are preserved!

To emphasize this symmetry, define for arbitrary, distinct vertices p and q of a protection graph G

*The names are not at all suggestive, however.

can·tell(p,q,G) to be true if and only if there is a sequence of protection graphs $G_1,\ldots,G_n$ such that G_{i+1} follows from G_i by application of one of these new rules or the DJ rules $(0 \le i < n)$ and in G_n there is a p to q edge labeled w.

Then we have from Theorem 5.1.

Corollary 6.1: *can·tell*(p,q,G) is true if and only if

(i) *can·share*(w,p,q,G) is true or,

(ii) there exists a sequence of subjects $u_1,\ldots,u_n$ such that the following conditions hold:

(a) $p = u_1$ or u_1 wr-initially spans to p,

(b) $q = u_n$ or u_n wr-terminally spans to q, and

(c) for all i, $1 \le i < n$ there is an rwtg-path between u_j and u_{j+1} with associated word in $B \cup C'$,

where wr-initial or wr-terminal spans are defined by interchanging r and w in the definitions of rw-initial and rw-terminal spans respectively and $C' = \{\vec{t}^{*}\vec{w} \cup \overleftarrow{r}\overleftarrow{t}^{*} \cup \vec{t}^{*}\vec{w}\overleftarrow{r}\overleftarrow{t}^{*}\}$. Of course, *can·tell·f*(p,q,G) can be similarly defined.

The second issue is our "worst-case" assumption. We have assumed perfect cooperation throughout this paper. It may be a prudent assumption but perhaps it is not very realistic. This assumption can be relaxed at the cost of further analysis in a way analogous to the way *can·share* was relaxed to *can·steal* in [3,4]. There, the owners of the information are assumed not to cooperate while all other subjects do. Alternatively, the number of cooperating subjects required for a transfer, called conspirators in [3], could be counted. This number could then be used as a measure of the probability that the transfer would actually be effected since a large number of collaborators are likely to be more difficult to enlist then a small number. The problem requries further study.

7. *References*

[1] A. K. Jones, R. J. Lipton and L. Snyder.
A linear-time algorithm for deciding security.
Proceedings of the 17th Annual Symp. on Foundations of Computer Science, 1976.

[2] R. J. Lipton and L. Snyder.
A linear-time algorithm for deciding subject security.
JACM 24(3):455-464, 1977.

[3] L. Snyder.
Theft and Conspiracy in the Take-Grant Protection Model.
Yale Department of Computer Science Technical Report #147, Nov. 1978.

[4] L. Snyder.
Synthesis and Analysis of Protection Systems.
Proceedings of the 6th Symp. on Operating Systems Principles, 1977.

[5] *The Concise Oxford Dictionary*, Oxford University Press, Sixth Edition, 1976.

OFFICIAL DISTRIBUTION LIST

Defense Documentation Center Cameron Station Alexandria, VA 22314	12 copies
Office of Naval Research Arlington, VA 22217	
Information Systems Program (437)	2 copies
Code 200	1 copy
Code 455	1 copy
Code 458	1 copy
Office of Naval Research Branch Office, Boston Bldg 114, Section D 666 Summer Street Boston, MA 02210	1 copy
Office of Naval Research Branch Office, Chicago 536 South Clark Street Chicago, IL 60605	1 copy
Office of Naval Research Branch Office, Pasadena 1030 East Green Street Pasadena, CA 91106	1 copy
Naval Research Laboratory Technical Information Division, Code 2627 Washington, D.C. 20375	6 copies
Dr. A. L. Slafkosky Scientific Advisor Commandant of the Marine Corps (Code RD-1) Washington, D.C. 20380	1 copy
Naval Ocean Systems Center Advanced Software Technology Division Code 5200 San Diego, CA 92152	1 copy
Mr. E. H. Gleissner Naval Ship Research and Development Center Computation and Mathematics Department Bethesda, MD 20084	1 copy

Captain Grace M.Hopper (008) 1 copy
Naval Data Automation Command
Washington Navy Yard
Building 166
Washington, D.C. 20374

Defense Advanced Research Projects Agency 3 copies
Attn: Program Management/MIS
1400 Wilson Boulevard
Arlington, VA 22209